



# INLS 490-154W: Information Retrieval Systems Design and Implementation

Fall 2009 Web-based course

**Query processing and retrieval**  
*Mapping information need to information*

**Chirag Shah**

School of Information & Library Science (SILS)

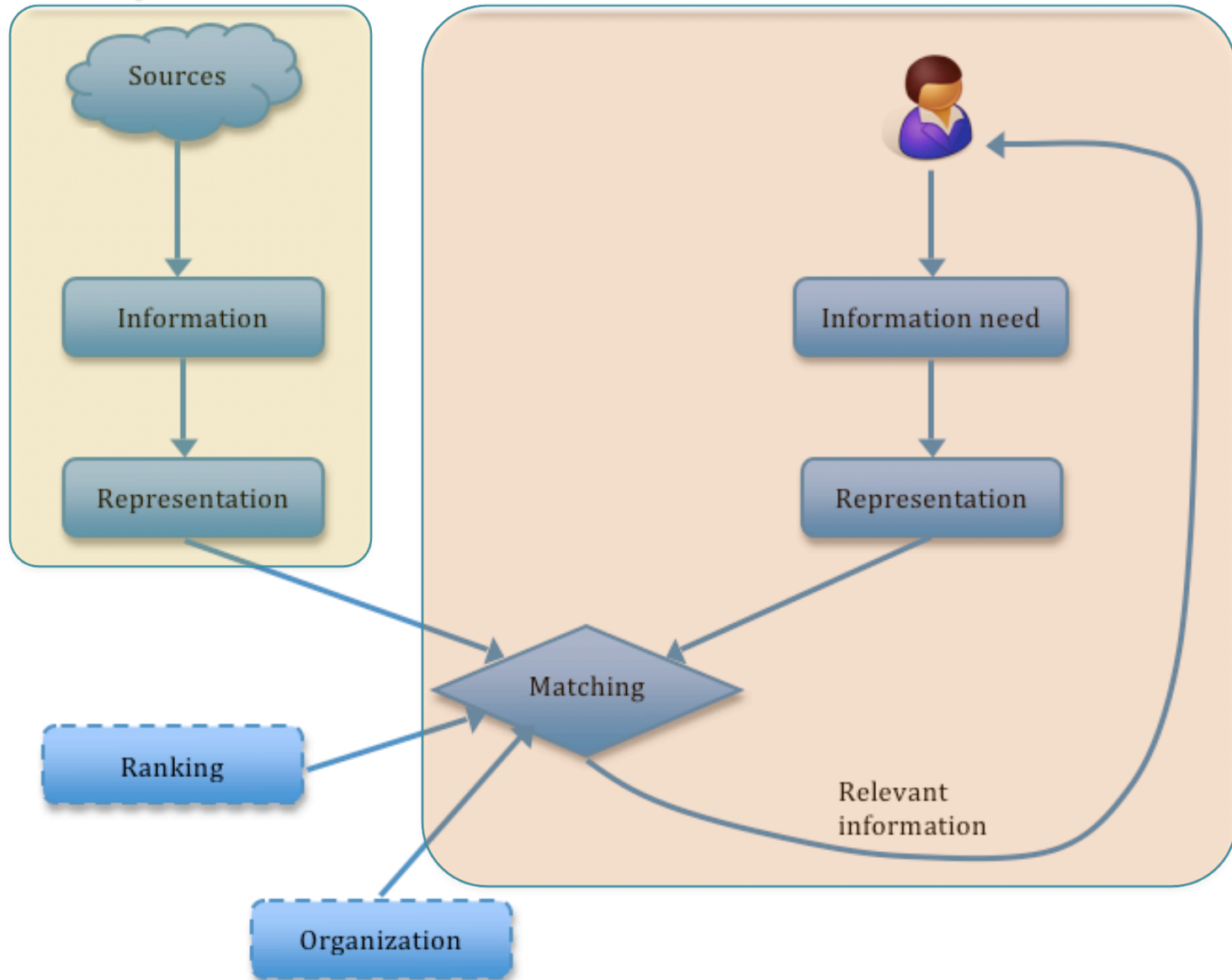
UNC Chapel Hill

---

## Last time

- Indexing = information representation
- IR = matching information need to information
- Steps for typical indexing:
  1. Collecting information
  2. Tokenization
  3. Stop words removal
  4. Stemming
  5. Storage

# Today's lab



# Indexing with Lemur

## BuildIndex <param\_file>

```
<parameters>  
  <index>myindex</index>  
  <indexType>indri</indexType>  
  <dataFiles>files.list</dataFiles>  
  <docFormat>trec</docFormat>  
  <stopwords>stopwords.list</stopwords>  
  <stemmer>krovetz</stemmer>  
</parameters>
```

# Analyzing the index

**dumpindex** <index\_name> <options>

- **s** (statistics)
  - Index statistics
- **v** (vocabulary)
  - <term> <term\_count> <doc\_count>
- **tp** <term> (term positions)
  - <term> <stem> <count> <total\_count>
  - <doc\_no> <count> <positions>
- **dt** <doc\_no> (document text)
  - Document content

# Indexing revisited

1. Collection
2. Tokenization
3. Stop words removal
4. Stemming
5. **Storage**

## ParseToFile <param\_file> <input\_files>

```
<parameters>
```

```
  <outputFile>parsed.txt</outputFile>
```

```
  <docFormat>trec</docFormat>
```

```
  <stopwords>stopwords.list</stopwords>
```

```
  <stemmer>krovetz</stemmer>
```

```
</parameters>
```

# Parsing a query

## ParseToFile <param\_file> <query\_file>

```
<parameters>  
  <outputFile>query_parsed.txt</outputFile>  
  <docFormat>trec</docFormat>  
  <stopwords>stopwords.list</stopwords>  
  <stemmer>krovetz</stemmer>  
</parameters>
```

# Retrieval

## RetEval <param\_file>

```
<parameters>
  <index>myindex</index>
  <textQuery>query_parsed.txt</textQuery>
  <resultFile>results.txt</resultFile>
  <resultFormat>3col</resultFormat>
  <resultCount>5</resultCount>
  <retModel>tfidf</retModel>
</parameters>
```

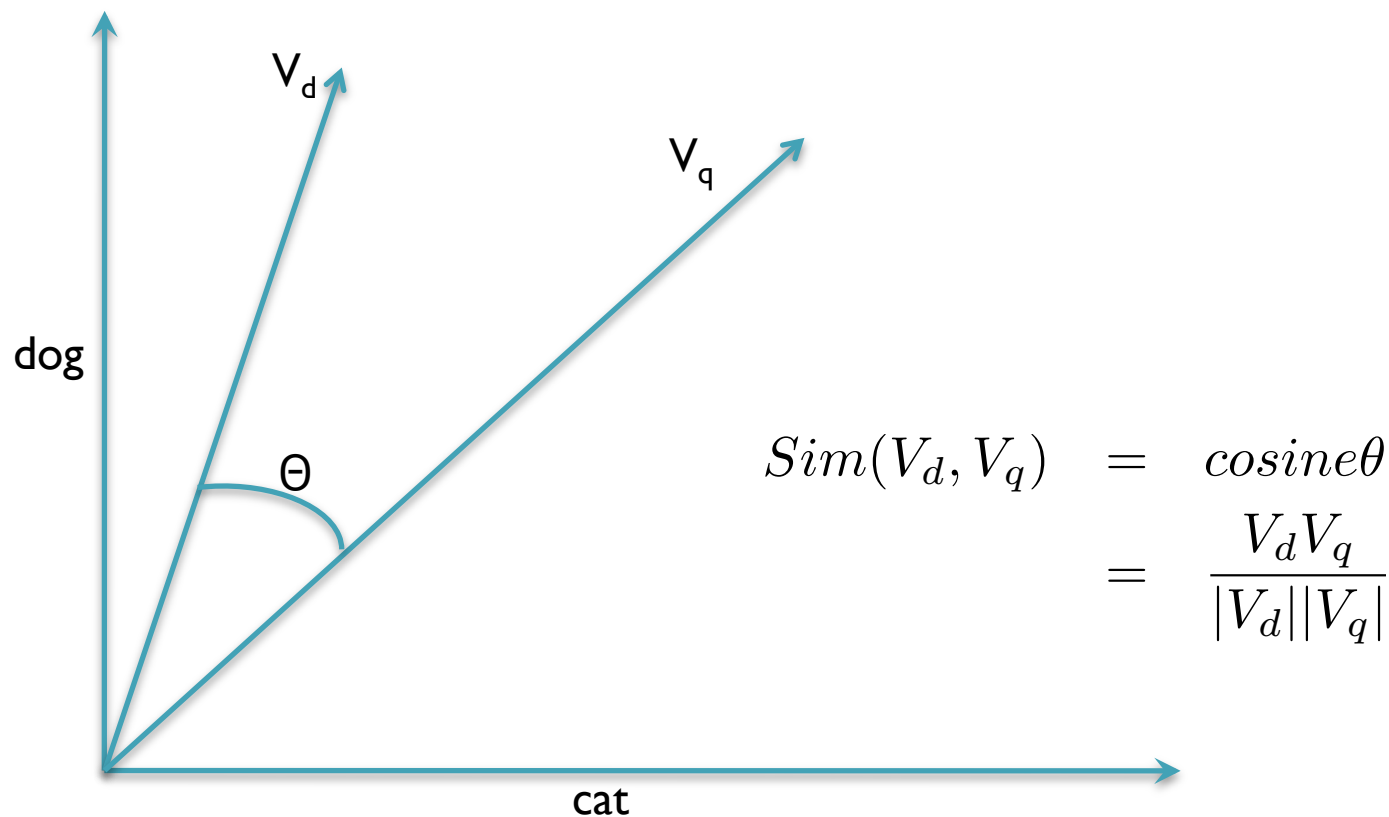
## Vector space model

- Key idea: a piece of information (document or query) is a vector in high-dimensional space

Vocab = {dog, cat}

$V_d = \{5, 2\}$

$V_q = \{1, 1\}$



## Term weighing

How “important” is a term?

TFIDF intuitions:

- (1) More frequency in a document = more important
- (2) Less common in a collection = more important

$$TF(t_i, d_j) = \frac{freq(t_i, d_j)}{\sum_k freq(t_i, d_k)}$$

$$IDF(t_i) = \log \frac{|D|}{|d_j : t_i \in d_j|}$$

$$TFIDF(t_i, d_j) = TF(t_i, d_j)IDF(t_i)$$

## Summary

- IR = matching information need to information
- For Web IR with textual information,
  - Information = text documents
  - Information need = text query
  - Results = set of text documents
- TFIDF is the most common way of weighing terms in IR
- Concepts covered: indexing, query processing, vector space model, TFIDF term weighing, cosine matching

# Next time

